

Vorwort

Seit einigen Jahren gibt es Qualitätsmanagement- und Qualitätssicherungsnormen für die Entwicklung, Lieferung und Wartung von Software (z. B. DIN ISO 9000–3) und ein fünfstufiges Reifegradmodell zur Bewertung des Entwicklungsprozesses einer Institution. Viele Software-Firmen und -Abteilungen haben sich seitdem ihren normgerechten Entwicklungsstandard zertifizieren lassen. Dennoch hat es keinen bemerkenswerten Sprung in der Qualität und Fehlerfreiheit der Softwareprodukte gegeben. Dies liegt vor allem daran, daß die oben zitierte DIN nur die Qualität des Entwicklungsprozesses und nicht die Qualität der erstellten *Produkte* im Auge hat und daß die meisten Institutionen noch nicht die beiden obersten Stufen des Reifegradmodells erreicht haben, bei denen erstmals quantitative und qualitative Produkteigenschaften betrachtet werden.

Daher konzentriert sich dieses Buch auf die Überprüfung entsprechender Produkteigenschaften, d. h. es werden Methoden und Verfahren vorgestellt, die Abweichungen (Fehler) aufdecken und beseitigen können. Die grundlegenden Konzepte werden durch entsprechende Problemstellungen motiviert, meistens formal definiert und umgangssprachlich und anhand von Beispielen und Abbildungen erläutert. Außerdem werden Angaben zu Vor- und Nachteilen (Kosten und Nutzen) der Testmethoden gemacht, mögliche Algorithmen werden vorgestellt und Eigenschaften der Verfahren werden bewiesen, wenn das im gegebenen Rahmen möglich ist, ansonsten wird auf entsprechende Quellen verwiesen.

Leserinnen und Leser — seien es Studierende an Universitäten oder Fachhochschulen, Personen in der Forschung oder in der Praxis (Software-Entwicklung oder -Management) — können dieses Buch daher zum Selbststudium verwenden und die dargestellten Methoden als Grundlage für die (Weiter-)Entwicklung entsprechender Werkzeuge verwenden. (Auf existierende Werkzeuge und Prototypen wird in diesem Buch nicht eingegangen, da dies den Rahmen sprengen würde und Werkzeuge schneller veralten als Methoden.)

Das Buch gliedert sich in vier Teile. Der einführende Teil I beginnt mit zwei Beispielen für den Programmtest. Es schließen sich zwei Kapitel an, die einen Überblick über die grundlegenden Problemstellungen und Lösungsansätze beim Testen geben. Teil II stellt in Kapitel 4 die „klassischen“ unsystematischen, datenbereichsbezogenen und funktionsbezogenen Testmethoden vor. Neuere Konzepte zum Testen auf der Basis von Pfadausdrücken und algebraischen Spezifikationen enthält Kapitel 5. Kapitel 6 widmet sich dem Vergleich und der Bewertung dieser Testmethoden. Die Methoden in Teil II kommen überwiegend beim spezifikationsorientierten Testen

vor. Sie können aber teilweise (in abgewandelter Form) auch beim implementationsorientierten Testen verwendet werden, das in Teil III abgehandelt wird. Dazu gehören Testmethoden, die sich am Kontrollfluß (Kapitel 7), am Datenfluß (Kapitel 8) oder an den Ausdrücken, Anweisungen und Daten (Kapitel 9) orientieren. Die Kapitel 10 und 11 vergleichen Kosten und Nutzen dieser Testmethoden.

Weitere Aspekte des Testens werden in Teil IV behandelt. In Kapitel 12 werden die — früh einsetzbaren — Techniken der statischen Analyse und die bisher nur theoretisch interessanten Techniken der symbolischen Ausführung und Verifikation vorgestellt. Kapitel 13 behandelt das praktisch relevante Testen von großen Programmsystemen durch Integrations- und Systemtests. Kapitel 14 stellt die neuartigen Probleme und Lösungsansätze zum Testen nebenläufiger Systeme vor. Da das Testen nur eine Diagnose liefert, die Therapie in Form der Fehlerbehebung aber das eigentliche Ziel ist, werden entsprechende Techniken in Kapitel 15 vorgestellt. Das Management des Testens auf der Grundlage von Komplexitätsmaßen für die (Test-)Aufwandsermittlung und die Auswahl von Testmethoden, die kombinierte Verwendung von Testmethoden sowie die Abschätzung der Restfehler und der Zuverlässigkeit sind Inhalte von Kapitel 16. Kapitel 17 faßt die Ergebnisse des Buches kurz zusammen und gibt einen Ausblick auf neue Herausforderungen für Softwaretestmethoden. Für weitergehende Studien ist eine Aufstellung von einschlägigen Zeitschriften und Standards sowie ein umfangreiches Literaturverzeichnis im Anhang zu finden.

Dozenten können jeden der vier Teile dieses Buches in ca. 24 Stunden präsentieren (z. B. im Sommersemester mit 12 Wochen à zwei Stunden pro Woche), wobei Teil I zusammen mit Kapitel 12 vorgetragen werden sollte.

Dieses Buch wurde aus mehreren Quellen gespeist. Aufgrund von Anregungen durch meine einjährige Tätigkeit in einem Dortmunder Softwarehaus habe ich im Wintersemester 1984/85 das Thema Softwaretesten zum Inhalt einer Vorlesung an der Universität Dortmund gemacht, zu der mein Kollege Herbert Schippers die Übungen beisteuerte. In den folgenden Jahren haben wir weitere Erfahrungen mit dem Einsatz und der Entwicklung von Testmethoden und Testwerkzeugen in Projekten, studentischen Projektgruppen und Diplomarbeiten gemacht. Auf Anregung des Teubner-Verlags entwickelten Herbert Schippers und ich Ende 1989 ein Konzept für eine erste Version eines Buches auf Grundlage des Vorlesungsskripts „Softwaretestmethoden“. Herbert Schippers hat die erste Fassung der Kapitel 3 und 4 geschrieben, die anderen Kapitel wurden von mir beigesteuert und als Grundlage für entsprechende Vorlesungen vom Wintersemester 1990/91 bis zum Sommersemester 1996 verwendet und parallel dazu von mir überarbeitet und ergänzt. Für die vorliegende Fassung des Textes trage ich — nach dem Ausscheiden von Herbert Schippers aus dem „Buchprojekt“ Ende 1990 — die alleinige Verantwortung.

Zu Stil und Rechtschreibung dieses Buches sind fünf Anmerkungen angebracht:

1. Die verwendeten Begriffe werden meistens in (numerierten) Definitionen explizit eingeführt oder aber durch Fettdruck hervorgehoben. Diese „Definitionen“ sind nicht immer wie formale Definitionen im mathematischen Sinne zu verstehen, sondern oft als informelle, möglichst konkrete Begriffsbeschreibungen zu sehen. Ebenso verhält es sich mit Aussagen über Testmethoden, die teilweise als (numerierte) „Sätze“ formuliert sind.
2. Da dieses Buch in deutscher Sprache geschrieben ist, werden — wenn immer möglich — deutsche Begriffe verwendet. Bei unüblichen deutschen Begriffen wird die englische Bezeichnung als Erläuterung in Klammern hinzugefügt.
3. Zur Behandlung des Problems mit den „männlichen“ Berufsbezeichnungen gibt es bisher mehrere Ansätze:
 - (a) einmaliger Hinweis darauf, daß z. B. mit „Softwaretester“ auch Frauen gemeint sind,
 - (b) konsequente Verwendung der männlichen und weiblichen Form (z. B. Softwaretester und Softwaretesterin),
 - (c) neue Form mit großem „I“ (z. B. SoftwaretesterIn),
 - (d) Verwendung der rein weiblichen Form.

Da diese Ansätze alle nicht befriedigen, habe ich mich für einen neuen Ansatz entschieden: In den Kapiteln 2 und 12 wird die rein weibliche oder eine neutrale Form, in den anderen Kapiteln nur die männliche oder eine neutrale Form verwendet.

4. Hervorhebungen werden folgendermaßen verwendet: **Fettdruck** für definierte Begriffe, *Kursivdruck* für die Betonung einzelner Wörter (als Lesehilfe) bzw. durchgängig zur Abgrenzung von Beispielen, Unterstreichungen als Gliederungshilfe, um Unterpunkte abzugrenzen, und in durchgängig kursiv geschriebenen Texten als Hervorhebung einzelner Wörter.
5. Die Rechtschreibung orientiert sich noch nicht an den neuen Rechtschreibregeln von 1997, daher wird z. B. das Wort „Testen“ in der Form „Te–sten“ und nicht als „Tes–ten“ getrennt.

Dieses Buch wäre nicht zustande gekommen ohne die wohlwollende Unterstützung und Duldung durch meinen unmittelbaren Vorgesetzten, Prof. Dr. Bernd Reusch, und meine Familienmitglieder Bärbel, Sonja und Inga, die mir erlaubt haben, einen Teil meiner Arbeits- und Freizeit für dieses Buch zu verwenden. Zu danken habe ich auch dem Herausgeber, Prof. Dr. Volker Claus, und dem Lektor im Teubner-Verlag, Dr. Peter Spuhler, für ihre Anregungen, ihre Geduld und insbesondere für

ihr Verständnis für die Verzögerungen bei der Erstellung des Buches.

Vielen meiner (Ex-)Kolleginnen und Kollegen habe ich für ihre kritischen und konstruktiven Bemerkungen und Hinweise auf Fehler in einzelnen Kapiteln von Vorversionen des Buchtextes zu danken, insbesondere Wolfgang Ditt, Beate Fricke, Arnulf Mester und meinem GI-Kollegen Andreas Spillner. Ebenso danke ich meinen Diplomanden und den Hörerinnen und Hörern meiner Vorlesungen, André Deparade, Nicola Engel und Martin Pfeiffer, insbesondere aber Norbert Feldhaus und Uwe Tegethoff. Für die letzte Version des vollständigen Buchtextes haben sich dankenswerterweise Uli Mette und Andreas Wieck als Korrekturleser angeboten.

Eine Vorversion des Textes hat Uwe Halfter angefertigt (in Wordplus auf einem Atari). Die endgültige Version des Textes und die Bilder wurden von Thomas Biskup in den letzten drei Jahren in unermüdlichem, nicht mit Geld zu bezahlendem Einsatz erstellt, außerdem hat er mit einer Fülle von kritischen Anmerkungen und Vorschlägen zur erheblichen Verbesserung der Qualität des Buches beigetragen.

Die Texte für dieses Buch wurden unter den Betriebssystemen Linux bzw. SunOS mit den Editoren „GNU Emacs“ und Jove in Form von 21 Dateien erstellt, außerdem wurden mit Xfig 93 Dateien mit Abbildungen und Tabellen erzeugt. Der Platzbedarf dieser 114 Dateien beträgt ca. 2,2 MByte. Für das Sachverzeichnis wurde von Thomas Biskup ein eigenes Programm namens **MakeIndex** (in perl) geschrieben, da das existierende Programm (für die Behandlung deutscher Umlaute) regelmäßig mit einem Fehler (segmentation fault) vorzeitig beendet wurde.

Mit Hilfe von zwei Dateien mit allgemeinen Formatierungsdefinitionen (teub_def.tex und zusatz.tex) wurde daraus mit dem Übersetzer für L^AT_EX₂e eine Datei im dvi-Format erzeugt, die anschließend in das postscript-Format umgewandelt und auf einem Laser-Drucker ausgedruckt wurde.

Das „Testproblem“ für einen Text ist mindestens so schwierig wie für ein Programm, da die benutzten Textverarbeitungswerkzeuge mehr Fehler durchgehen lassen als die Übersetzer (Compiler) von Programmiersprachen. Einige Tippfehler oder auch gedankliche Fehler werden das Korrekturlesen (die „Text-Inspektion“, vgl. Kapitel 12) vermutlich ebenfalls überlebt haben, obwohl viele „Inspekteure“ (s. oben) beteiligt waren. Ich bin daher — wie die potentiellen Leser der nächsten Auflage dieses Buches — dankbar für jeden Korrekturvorschlag an meine Dienstadresse:

Universität Dortmund, Informatik I, Postfach 500 500, 44221 Dortmund
E-Mail: riedeman@ls1.informatik.uni-dortmund.de

Dortmund, im Juli 1997

Eike Hagen Riedemann

Inhalt

I	Einführung	15
1	Beispiele für den Programmtest	17
1.1	Selbsttest	17
1.2	Beispiel für ein sequentielles Programm	19
1.3	Beispiel für ein nebenläufiges Programm	19
2	Grundlegende Problemstellungen und Lösungsansätze	21
2.1	Ursachen unangemessener und fehlerhafter Software	21
2.2	Manifestation von Softwarefehlern	25
2.3	Fehlverhalten von Software und seine Kosten	31
2.4	Vermeidung und Behebung von Fehlern	34
2.5	Kosten und Nutzen des Testens	39
2.6	Leben mit Fehlern	41
3	Qualitätsmanagement-, Prüf- und Testmethoden im Überblick	47
3.1	Ziel, Intention und Vorgehensweise des Testens	47
3.2	Einordnung des Gebiets Testen in das Qualitätsmanagement	49
3.3	Statische und dynamische Prüfverfahren	53
3.4	Grundsätzliche Teststrategien und Testansätze	56
3.5	Testablauf	60
3.6	Testphasen im Software-Entwicklungsprozeß	64
3.7	Überprüfung des Entwicklungs- und Qualitätsmanagementsystems	66
3.8	Testmanagement	68
3.9	Verwendete Quellen und weiterführende Literatur	69
	Zusammenfassung von Teil I	70

II	Spezifikationsorientiertes Testen	71
4	Datenbereichsbezogenes und funktionsbezogenes Testen	74
4.1	Unsystematisches datenbereichs- oder funktionsbezogenes Testen . . .	74
4.2	Systematisches datenbereichsbezogenes Testen	75
4.3	Systematisches funktionsbezogenes Testen	99
4.4	Übungen	111
4.5	Verwendete Quellen und weiterführende Literatur	112
5	Testen von Reihenfolgebedingungen und algebraischen Spezifikationen	113
5.1	Testen von Reihenfolgebedingungen	113
5.2	Testen auf der Basis von algebraischen Spezifikationen	121
5.3	Übungen	135
5.4	Verwendete Quellen und weiterführende Literatur	137
6	Bewertung des spezifikationsorientierten Testens	138
6.1	Enthaltensein und Unvergleichbarkeit von Testkriterien	139
6.2	Anzahl der Testdaten pro Testkriterium	144
6.3	Aufgedeckte Fehler pro Testkriterium	151
6.4	Testdatenerzeugung und Messung der Testwirksamkeit	161
6.5	Übungen	179
6.6	Verwendete Quellen und weiterführende Literatur	182
	Zusammenfassung von Teil II	183
III	Implementationsorientiertes Testen	185
7	Kontrollflußbezogenes Testen	188
7.1	Kontrollflußgraphen und ihre Eigenschaften	188
7.2	Methoden des kontrollflußbezogenen Testens	194
7.3	Übungen	208
7.4	Verwendete Quellen und weiterführende Literatur	211

8	Datenflußbezogenes Testen	212
8.1	Problemstellung und Modellbildung	212
8.2	Einfache Datenflußkriterien	215
8.3	Verkettung von Datenflüssen	221
8.4	Parallele Betrachtung von Datenflüssen	224
8.5	Übungen	226
8.6	Verwendete Quellen und weiterführende Literatur	228
9	Ausdrucks-, anweisungs- und datenbezogenes Testen	229
9.1	Ausdrucks- und anweisungsbezogenes Testen	229
9.2	Test Boolescher Ausdrücke	236
9.3	Mutationsanalyse	244
9.4	Datenbezogenes Testen	246
9.5	Übungen	247
9.6	Verwendete Quellen und weiterführende Literatur	249
10	Bewertung der implementationsorientierten Testkriterien	251
10.1	Enthaltensein und Unvergleichbarkeit von Testkriterien	251
10.2	Anzahl der Testdaten pro Testkriterium	257
10.3	Aufgedeckte Fehler pro Testkriterium	260
10.4	Übungen	272
10.5	Verwendete Quellen und weiterführende Literatur	273
11	Testdatenerzeugung und Testwirksamkeitsmessung	274
11.1	Messung der Testwirksamkeit durch Instrumentierung	274
11.2	Testdatenerzeugung	282
11.3	Übungen	291
11.4	Verwendete Quellen und weiterführende Literatur	294
	Zusammenfassung von Teil III	295

12	Inhalt	
IV	Weitere Aspekte des Testens	297
12	Statische Analyse und symbolische Ausführung	300
12.1	Informelle Analyse	301
12.2	Formale Analyse	313
12.3	Symbolische Ausführung	327
12.4	Formale Verifikation	333
12.5	Übungen	339
12.6	Verwendete Quellen und weiterführende Literatur	340
13	Testen „im Großen“	342
13.1	Probleme beim Testen großer Systeme	342
13.2	Modelle für große Systeme	345
13.3	Strategien für den Modul- und Integrationstest	348
13.4	Aufwand, Fehlerarten und Voraussetzungen des Integrationstests	360
13.5	Testmethoden für den Integrationstest	363
13.6	System- und Abnahmetest, Installations- und Wartungsphase	371
13.7	Übungen	372
13.8	Verwendete Quellen und weiterführende Literatur	376
14	Testen nebenläufiger Systeme	378
14.1	Eigenschaften nebenläufiger Systeme und auftretende Fehler	378
14.2	Grundsätzliche Modelle, Probleme und Lösungen	384
14.3	Statische Analyse nebenläufiger Programme	395
14.4	Dynamische Analyse nebenläufiger Programme	400
14.5	Analyse von verteilten und Echtzeitsystemen	404
14.6	Übungen	411
14.7	Verwendete Quellen und weiterführende Literatur	413
15	Fehlerlokalisierung und -korrektur	417
15.1	Problemstellung, Lösungsstrategien und Wissensarten	417
15.2	Methoden der Fehlerlokalisierung	419
15.3	Prinzipien der Fehlerkorrektur und Fehleranalyse	435
15.4	Übungen	437
15.5	Verwendete Quellen und weiterführende Literatur	438

16	Management des Testens und Prüfens	439
16.1	Komplexitätsmaße für die Aufwandsermittlung	439
16.2	Auswahl von Softwareprüfmethoden	445
16.3	Kombination von Softwareprüfmethoden	452
16.4	Abschätzung von Fehleranzahl und Zuverlässigkeit	453
16.5	Spezielle Managementfragen	458
16.6	Übungen	460
16.7	Verwendete Quellen und weiterführende Literatur	461
17	Zusammenfassung und Ausblick	463
17.1	Zusammenfassung	463
17.2	Ausblick	463
17.3	Verwendete Quellen und weiterführende Literatur	466
A	Lösungen zu den Testaufgaben	467
A.1	Lösung zu Testaufgabe 1	467
A.2	Lösung zu Testaufgabe 2	468
A.3	Lösung zu Testaufgabe 3	471
A.4	Lösung zu Testaufgabe 4	472
B	Organisationen, Konferenzen, Zeitschriften und Standards	473
	Literatur	477

